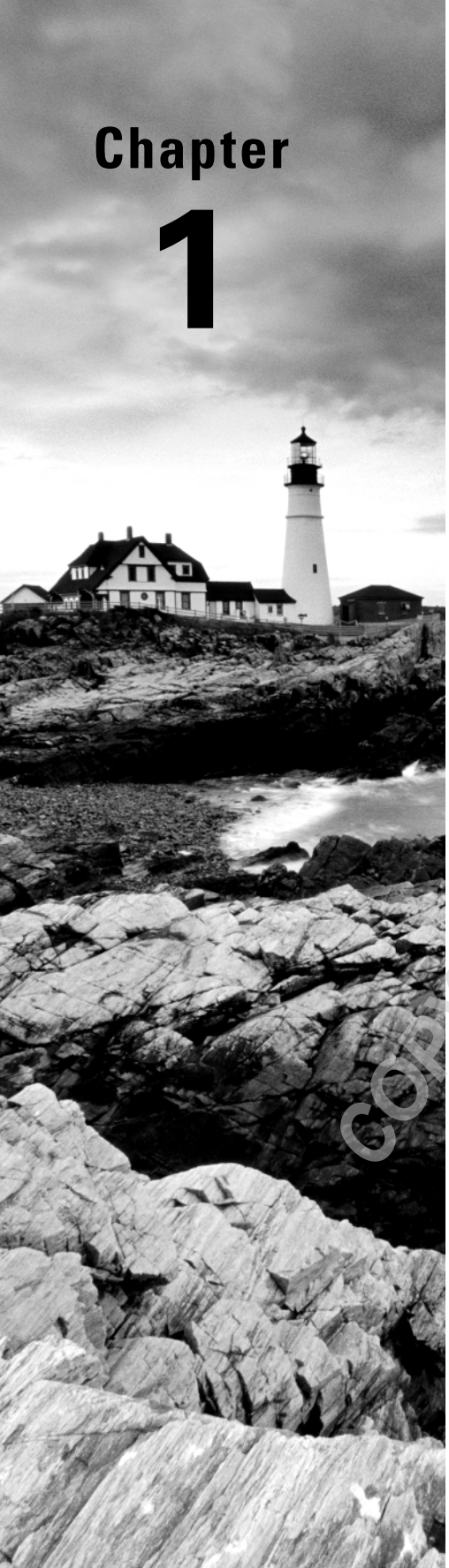


Chapter

1

Selecting an Operating System

OBJECTIVES:

- ✓ 1.1 Linux Evolution and Popular Operating Systems
- ✓ 4.1 Choosing an Operating System

The fact that you're reading this book means you want to learn about the Linux operating system (OS). To begin this journey, you must first understand what Linux is and what an OS is. This chapter describes what an OS is, how users interact with an OS, how Linux compares to other popular OSs, and how even specific Linux implementations vary. Understanding these issues will help you as you make the switch to Linux and learn about the various Linux-based systems.

What Is an OS?

An OS provides all the fundamental features of a computer, at least from a software point of view. An OS enables you to use the computer's hardware devices, defines the user interface standards, and provides basic tools that allow applications to run on the computer. This section describes the different parts that make up an OS and how they work together to create your computing experience.

What Is a Kernel?

An OS *kernel* is a software component responsible for managing various low-level features of the computer, including:

- Interfacing with hardware devices (network adapters, hard disks, and so on)
- Allocating memory to individual programs
- Allocating CPU time to individual programs
- Enabling programs to interact with one another

When you use a program (say, a web browser), it relies on the kernel for many of its basic functions. The web browser can communicate with the outside world only by using network functions provided by the kernel. The kernel allocates memory and CPU time to the web browser, without which it couldn't run. The web browser may rely on plug-ins to display multimedia content; such programs are launched and interact with the web browser through kernel services. Any program you run on a computer relies on the kernel in a similar way, although the details vary from one OS to another and from one program to another.

The kernel is the software "glue" that holds the computer together. Without a kernel, a modern computer can do very little.

Kernels are not interchangeable; the Linux kernel is different from the macOS kernel used in Apple workstations and laptops, and from the Windows kernel used in Microsoft-compatible workstations and laptops. Each of these kernels uses a different internal design and provides different software interfaces for programs to use. Thus, each OS is built from the kernel up and uses its own set of programs that further define each OS's features.

Some programs run on multiple kernels, but most need OS-specific tweaks. Programmers create *binaries*—the program files for a particular processor and kernel—for each OS. You need to run the binary file created for the specific OS you're running the program on.

Linux uses a kernel called *Linux*—in fact, technically speaking, the word Linux refers only to the kernel. Other features that you might associate with Linux are provided by non-kernel programs, most of which are available on other platforms, as described shortly, in "What Else Identifies an OS?"

A student named Linus Torvalds created the Linux kernel in 1991. Linux has evolved considerably since that time. Today, it runs on a wide variety of CPUs and other hardware. The easiest way to learn about Linux is to use it on a desktop or laptop PC, so that's the type of configuration emphasized in this book. The Linux kernel, however, runs on everything from tiny cell phones to powerful supercomputers.

What Else Identifies an OS?

The kernel is at the core of any OS, but it's a component that most users don't directly manipulate. Instead, most users interact with a number of other software components, many of which are closely associated with particular OSs. Such programs include the following:

Command-Line Shells Years ago, users interacted with computers exclusively by typing commands in a program (known as a *shell*) that accepted such commands. The commands would rename files, launch programs, and so on. Although many computer users today don't use text-mode shells, they're still important for intermediate and advanced Linux users, so we describe them in more detail in Chapter 5, "Getting to Know the Command Line," and subsequent chapters rely heavily on your ability to use a text-mode shell. Many different shells are available, and which shells are available and popular differ from one OS to another. In Linux, a shell known as the Bourne Again Shell (Bash) is popular.

Certification Objective

Graphical User Interfaces A graphical user interface (GUI) is an improvement on a text-mode shell, at least from the perspective of a beginning user. GUIs rely on icons, menus, and a mouse pointer rather than typed commands. Windows and macOS both have their own OS-specific GUIs. Linux relies on a GUI known as the X Window System, or X for short. X is a very basic GUI, so Linux also uses *desktop environment* program suites, such as the GNU Object Model Environment (GNOME) or the K Desktop Environment (KDE), to provide a more complete user experience. It's the differences between a Linux desktop

Certification Objective

environment and the GUIs in Windows or macOS that will probably strike you most when you first begin using Linux.

Utility Programs Modern OSs invariably ship with a wide variety of simple utility programs—calculators, calendars, text editors, disk maintenance tools, and so on. These programs differ from one OS to another. Even the names and methods of launching these programs can differ between OSs. Fortunately, you can usually find the programs you want by perusing menus in the main desktop environment.

Libraries Unless you're a programmer, you're unlikely to have to work with libraries directly; nonetheless, we include them in this list because they provide critical services to programs. Libraries are collections of programming functions that can be used by a variety of programs. For instance, in Linux most programs rely on a library called `libc`. Other libraries provide features associated with the GUI or that help programs parse options passed to them on the command line. Many libraries exist for Linux, which helps enrich the Linux software landscape.

Productivity Programs Major productivity programs—web browsers, word processors, graphics editors, and so on—are the usual reason for using a computer. Although such programs are often technically separate from the OS, they are sometimes associated with certain OSs. Even when a program is available on many OSs, it may have a different "feel" on each OS because of the different GUIs and other OS-specific features.

You can search for Linux equivalents to popular macOS or Windows programs on popular open source software websites such as `www.linuxalt.com`.

In addition to software that runs on an OS, several other features can distinguish between OSs, such as the details of user accounts, rules for naming disk files, and technical details of how the computer starts up. These features are all controlled by software that's part of the OS, sometimes by the kernel and sometimes by non-kernel software.

Investigating User Interfaces

Earlier, we noted the distinction between text-mode and graphical user interfaces. Although most users favor GUIs because of their ease of use, Linux retains a strong text-mode tradition. Chapter 5 describes Linux's text-mode tools in more detail, and Chapter 4, "Using Common Linux Programs," covers basic principles of Linux GUI operations. It's important that you have some grounding in the basic principles of both text-mode and graphical user interfaces now, since user interface issues crop up from time to time in intervening chapters.

Using a Text-Mode User Interface

Certification Objective

In the past, and even sometimes today, Linux computers booted in text mode. After the system had completely booted, the screen would display a simple text-mode login prompt, which might resemble this:

```
Fedora 30 (Workstation Edition)
Kernel 5.0.9-301.fc30.x86_64 on an x86_64 (tty1)

essentials login:
```

To try a text-mode login, you must first install Linux on a computer. Neither the Linux Essentials exam nor this book covers Linux installation; consult your distribution's documentation to learn more.

The details of such a login prompt vary from one system to another. This example includes several pieces of information:

- The OS name and version: Fedora Linux version 30
- The computer's name: `essentials`
- The name of the hardware device being used for the login: `tty1`
- The login prompt itself: `login:`

If you see a GUI login prompt, you can obtain a text-mode prompt by pressing Ctrl+Alt+F2 or Ctrl+Alt+F3. To return to the GUI login prompt, press Ctrl+Alt+F1 or Ctrl+Alt+F7.

To log into such a system, you must type your username at the `login:` prompt. The system then prompts you for a password, which you must also type. If you entered a valid username and password, the computer is likely to display a login message, followed by a shell prompt:

```
[rich@essentials:~]$
```

In this book, we omit most of the prompt from example commands when they appear on their own lines. We keep the dollar sign (`$`) prompt, though, for ordinary user commands. Some commands must be entered as the `root` user account, which is the Linux administrative user. We change the prompt to a hash mark (`#`) for such commands, since most Linux distributions make a similar change to their prompts for the `root` user.

Chapter 13, "Creating Users and Groups," describes Linux accounts, including the `root` user account, in more detail.

The details of the shell prompt vary from one installation to another, but you can type text-mode commands at the shell prompt. For instance, you could type **`ls`** (short for list) to see a list of files in the current directory. The most basic commands are shortened by removing vowels, and sometimes consonants, to minimize the amount of typing required to execute a command. This has the unfortunate effect of making many commands rather obscure.

Some commands display no information, but most produce some type of output. For instance, the `ls` command produces a list of files:

```
$ ls
chapter1.doc  figure01.png
```

This example shows two files in the current directory: `chapter01.doc` and `figure01.png`. You can use additional commands to manipulate these files, such as `cp` to copy them or `rm` to remove (delete) them. Chapter 5 ("Getting to Know the Command Line") and Chapter 7 ("Managing Files") describe some common file manipulation commands.

Some text-mode programs take over the display in order to provide constant updates or to enable you to interact with data in a flexible manner. Figure 1.1, for instance, shows the nano text editor, which is described in more detail in Chapter 10, "Editing Files." When nano is working, you can use your keyboard's arrow keys to move the cursor around, add text by typing, and so on.

FIGURE 1.1 Some text-mode programs take over the entire display.

Even if you use a graphical login, you can use a text-mode shell inside a window, known as a *terminal*. With common Linux GUIs, you can launch a terminal program, which provides a shell prompt and the means to run text-mode programs.

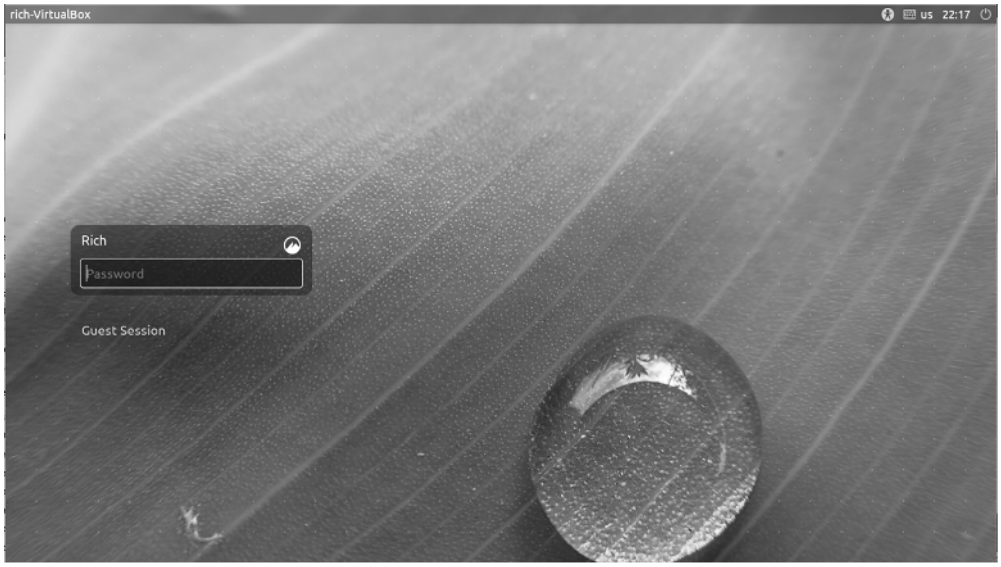
Using a Graphical User Interface

Certification Objective

Most users are more comfortable with GUIs than with text-mode commands. Thus, many modern Linux systems start up in GUI mode by default, presenting a login screen similar to the one shown in Figure 1.2. You can select your username from a list or type it, followed by typing your password, to log in.

Some Linux GUI login screens don't prompt you for a password until after you've entered a valid username.

FIGURE 1.2 Graphical login screens on Linux are similar to those for Windows or macOS.



Unlike Windows and macOS, Linux provides a number of different desktop environments for you to choose from. Which one you use depends on the specific variety of Linux you're using, what software options you selected at installation time, and your own personal preferences. Common choices include GNOME, KDE Plasma, Cinnamon, and Xfce. Many other options are available as well. Many graphical desktops have assistive technology features built in. In Figure 1.2, the person icon at the top-right corner of the Fedora login window allows you to select an assistive technology such as a screen reader or onscreen keyboard to assist with the login entry.

Linux desktop environments can look quite different from one another, but they all provide similar functionality. Figure 1.3 shows the default Cinnamon desktop on a Mint 18.3 installation, with a couple of programs running. Chapter 4 describes common desktop

environments and their features in more detail, but for now, you should know that they all provide features such as the following:

Program Launchers You can launch programs by selecting them from menus or lists. Typically, one or more menus reside along the top, bottom, or side of the screen. In Figure 1.3, you can click the Mint leaf icon in the bottom-left corner of the screen to produce the menu that appears in that figure.

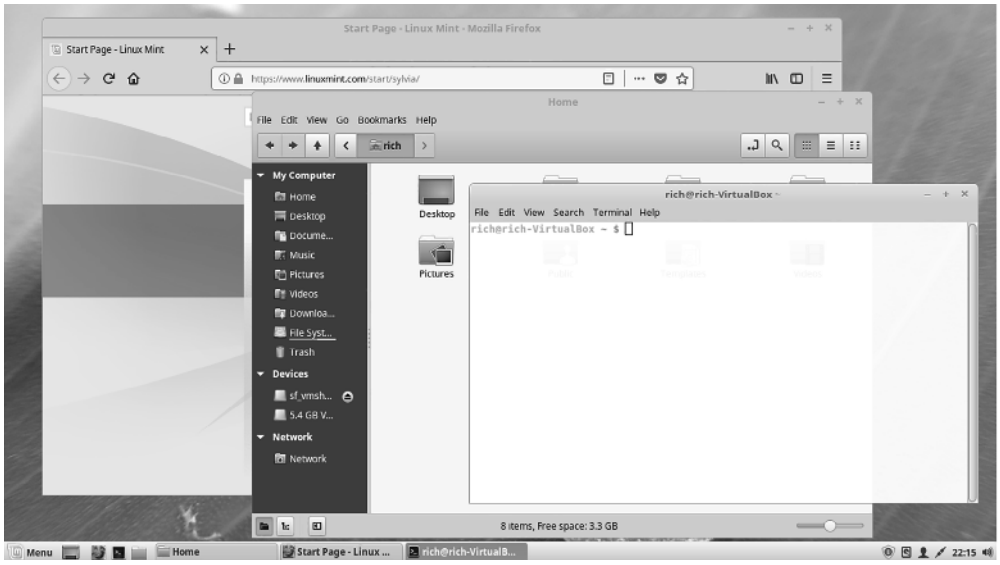
File Managers Linux provides GUI file managers similar to those in Windows or macOS. A window for one of these is open in the center of Figure 1.3.

Window Controls You can move windows by clicking and dragging their title bars, resize them by clicking and dragging their edges, and so on.

Multiple Desktops Most Linux desktop environments enable you to keep multiple virtual desktops active, each with its own set of programs. This feature helps keep the screen uncluttered while you run several programs simultaneously. Typically, an icon in one of the menus enables you to switch between virtual desktops.

Logout Options You can log out of your Linux session, which enables you to shut down the computer or let another user log in.

FIGURE 1.3 Linux desktop environments provide the types of GUI controls that most users expect.



Logging out is very important in public computing environments. If you fail to log out, a stranger might come along and use your account for malicious purposes.

As you learn more about Linux, you'll discover that its GUI environments are quite flexible. If you find you don't like the environment that's the default for your distribution, you can change it. Although they all provide similar features, some people have strong preferences about desktop environments. Linux gives you a choice in the matter that's not available in Windows or macOS, so feel free to try multiple desktop environments.

You may need to install extra desktop environments to use them. This topic is not covered in this book.

Where Does Linux Fit in the OS World?

This chapter's title implies a comparison, and as this book is about Linux, the comparison must be to non-Linux OSs. Thus, we compare Linux to three other OSs or OS families: Unix, Apple macOS, and Microsoft Windows.

As described later in "What Is a Distribution?" Linux can be considered a family of OSs. Thus, you can compare one Linux version to another one.

Comparing Linux to Unix

Certification Objective

If you were to attempt to draw a "family tree" of OSs, you would end up scratching your head a lot. This is because OS designers often mimic one another's features and sometimes even incorporate one another's code into their OSs' workings. The result can be a tangled mess of similarities between OSs, with causes ranging from coincidence to code "borrowing." Attempting to map these influences can be difficult. In the case of Linux and Unix, though, a broad statement is possible: Linux is modeled after Unix.

Unix was created in 1969 at AT&T's Bell Labs. Unix's history is complex and involves multiple *forks* (that is, splitting of the code into two or more independent projects) and even entirely separate code rewrites. Modern Linux systems are, by and large, the product of open source projects that clone Unix programs, or of original open source code projects for Unix generally.

Open source software is software that you can not only run, but modify and redistribute yourself. Chapter 3, "Investigating Linux's Principles and Philosophy," covers the philosophy and legal issues concerning open source software.

These projects include:

The Linux Kernel Linus Torvalds created the Linux kernel as a hobby programming project in 1991, but it soon grew to be much more than that. The Linux kernel was designed to be compatible with other Unix kernels in the sense that it used the same software interfaces in source code. This made using open source programs for other Unix versions with the Linux kernel easy.

The GNU Project The GNU's Not Unix (GNU) project is an effort by the Free Software Foundation (FSF) to develop open source replacements for all the core elements of a Unix OS. In 1991, the FSF had already released the most important such tools, with the notable exception of the kernel. (The GNU HURD kernel is now available but is not as popular as the Linux kernel.) Alternatives to the GNU tools include proprietary commercial tools and open source tools developed for the BSD Unix variants. The tools used on a Unix-like OS can influence its overall "flavor," but all of these tool sets are similar enough to give any Unix variety a similar feel compared to a non-Unix OS.

GNU is an example of a recursive acronym—an acronym whose expansion includes the acronym itself. This is an example of geek humor.

Xorg-X11 The X Window System is the GUI environment for most Unix OSs. Most Linux distributions today use the Xorg-X11 variety of X. As with the basic text-mode tools provided by the GNU project, choice of an X server can affect some features of a Unix-like OS, such as the types of fonts it supports. Wayland is a newer X Windows System software package used in Linux and is gaining in popularity.

Desktop Environments GNOME, KDE Plasma, Cinnamon, Xfce, and other popular open source desktop environments have largely displaced commercial desktop environments even on commercial versions of Unix. Thus, you won't find big differences between Linux and Unix in this area.

macOS, described shortly, is technically a commercial Unix but uses a proprietary Apply GUI instead of an open source desktop environment running on the X Window System.

Server Programs Historically, Unix and Linux have been popular as server OSs—organizations use them to run web servers, database servers, email servers, and so on. Linux runs the same popular server programs as do commercial Unix versions and the open source BSDs.

User Productivity Programs In this realm, as in server programs, Linux runs the same software as do other Unix-like OSs. In a few cases, Linux runs more programs or runs them better. This is mostly because of Linux's popularity and the vast array of hardware drivers that Linux offers. If a program needs advanced video card support, for example, it's more likely to find that support on Linux than on a less popular Unix-like OS.

On the whole, Linux can be thought of as a member of the family of Unix-like OSs. Although Linux is technically not a Unix OS, it's similar enough that the differences are unimportant compared to the differences between this family as a whole and other OSs, such as Windows. Because of its popularity, Linux offers better hardware support, at least on commodity PC hardware. Some Unix varieties offer specific features that Linux lacks, though. For instance, Oracle's Solaris Unix uses built-in zones that handle virtual machines better than the tools currently available in Linux.

Although computers understand only ones and zeroes, human beings prefer to write programs in a text form known as *source code*. Although source code can seem arcane to the uninitiated, it's crystal clear compared to the form a program must take for a computer to run it: *binary code*. A program known as a *compiler* translates source code to binary code. (Alternatively, some programming languages rely on an *interpreter*, which converts source code to binary code "on the fly," eliminating the need to compile source code.)

The term *open source* refers to the availability of source code, which is generally withheld from the public in the case of commercial programs and OSs. A programmer with access to a program's source code can fix bugs, add features, and otherwise alter how the program operates.

Comparing Linux to macOS

Certification Objective

Apple macOS is a commercial Unix-based OS that borrows heavily from the BSDs and discards the usual Unix GUI (namely X) in favor of its own user interface. This makes macOS both very similar to Linux and quite different from it.

You can open a macOS Terminal window and type many of the same commands described in this book to achieve similar ends. If a command described in this book isn't present, you may be able to install it in one way or another. macOS ships with some popular Unix server programs, so you can configure it to work much like Linux or another Unix-like OS as a network server computer.

macOS differs from Linux in its user interface, though. The macOS user interface is known as *Cocoa* from a programming perspective, or *Aqua* from a user's point of view. It includes elements that are roughly equivalent to both X and a desktop environment in Linux. Because Cocoa isn't compatible with X from a programming perspective, applications developed for macOS can't be run directly on Linux (or on other Unix-like OSs), and porting them (that is, modifying the source code and recompiling them) for Linux is a nontrivial undertaking. Thus, native macOS applications seldom make the transition to Linux.

macOS includes an implementation of X that runs under Aqua. This makes the transfer of GUI Linux and Unix programs to macOS relatively straightforward. The resulting programs don't entirely conform to the Aqua user interface, though. They may have buttons, menus, and other features that look out of place compared to the usual appearance of macOS equivalents.

Apple makes macOS available only for its own computers. Its license terms forbid installation on non-Apple hardware, and even aside from licensing issues, installing macOS on non-Apple hardware is a nontrivial undertaking. A variant of macOS, known as iOS, runs on Apple's iPad and iPhone devices, and is equally nonportable to other devices. Thus, macOS is largely limited to Apple hardware. Linux, by contrast, runs on a wide variety of hardware, including most PCs. You can even install Linux on Macintosh computers.

Comparing Linux to Windows

Certification Objective

Most desktop and laptop computers today run Microsoft Windows. Thus, if you're considering running Linux, the most likely comparison is to Windows. Broadly speaking, Linux and Windows have similar capabilities; however, there are significant differences in details. These include the following:

Licensing Linux is an open source OS whereas Windows is a proprietary commercial OS. Chapter 2, "Understanding Software Licensing," covers open source issues in greater detail, but for now you should know that open source software gives you greater control over your computer than does proprietary software—at least in theory. In practice, you may need a great deal of expertise to take advantage of open source's benefits. Proprietary software may be preferable if you work for an organization that's only comfortable with the idea of software that's sold in a more traditional way. (Some Linux variants, though, are sold in a similar way, along with service contracts.)

Costs Many Linux varieties are available free of charge and so are appealing if you're trying to cut costs. However, the expertise needed to install and maintain a Linux installation is likely to be greater, and therefore more expensive, than the expertise needed to install and maintain a Windows installation. Different studies on the issue of total cost of ownership of Linux versus Windows have gone both ways, but most tend to favor Linux.

Hardware Compatibility Most hardware components require OS support, usually in the form of drivers. Most hardware manufacturers provide Windows drivers for their devices or work with Microsoft to ensure that Windows includes appropriate drivers. Although some manufacturers provide Linux drivers, too, for the most part the Linux community as a whole must supply the necessary drivers. This means that Linux drivers may take a few weeks or even months to appear after a device becomes available. At the same time, Linux developers tend to maintain drivers for old hardware for much longer than manufacturers continue to support their own old hardware. Thus, a modern Linux may run better than a recent version of Windows on old hardware. Linux also tends to be less resource intensive, so you can be productive on older hardware when using Linux.

Software Availability Some popular desktop applications, such as Microsoft Office, are available on Windows but not on Linux. Although Linux alternatives such as LibreOffice are available, they haven't caught on in the public's mind. In other realms, the situation is reversed. Popular server programs, such as the Apache web server, were developed first for Linux or Unix. Although many such servers are available for Windows, they run more

efficiently on Linux. If you have a specific program you must run, you may want to research its availability and practicality on any platforms you're considering.

User Interfaces Like macOS, Windows uses its own unique user interface. This fact contributes to poor inter-OS portability. (Tools exist to help bridge the gap, though; X Window System implementations for Windows are available, as are tools for running Windows programs in Linux.) Some users prefer the Windows user interface to any Linux desktop environment, but others prefer a Linux desktop environment.

Microsoft introduced a new user interface, called Metro, in Windows 8. The idea behind Metro is that it works the same on everything from smartphones to desktop computers. However, the Metro user interface quickly became unpopular because it made it difficult for experienced desktop Windows users to maneuver around, so it was removed as the default user interface starting with Windows 10 but is still available for those who prefer it.

Configurability Linux is a much more configurable OS than is Windows. Although both OSs provide means to run specific programs at startup, change user interface themes, and so on, Linux's open source nature means you can tweak any detail you want. Furthermore, you can pick any Linux variant you like to get a head start on setting up the system as you see fit.

Security Advocates of each OS claim it's more secure than the others. They can do this because they focus on different security issues. Many of the threats to Windows come from viruses, which by and large target Windows and its huge installed user base. Viruses are essentially a nonissue for Linux; in Linux, security threats come mostly from break-ins involving misconfigured servers or untrustworthy local users.

For over a decade, Windows has dominated the desktop arena. In both homes and offices, users have become familiar with Windows and are used to popular Windows applications, such as Microsoft Office. Although Linux can be used in such environments, it's a less popular choice for a variety of reasons—its unfamiliarity, the fact that Windows comes preinstalled on most PCs, and the lack of any compelling Linux-only applications for most users.

Unix generally, and Linux in particular, on the other hand, have come to dominate the server market. Linux powers the web servers, database servers, email servers, and so on that make up the Internet and that many businesses rely on to provide local network services. Thus, most people use Linux daily even if they don't realize it.

In most cases, it's possible to use either Linux or Windows on a computer and have it do an acceptable job. Sometimes, though, specific needs dictate use of one OS or another. You might need to run a particular exotic program, for instance, or your hardware might be too old for a modern Windows or too new for Linux. In other cases, your own or your users' familiarity with one OS or the other may favor its use.

What Is a Distribution?

Up until now, we've described Linux as if it were a single OS, but this isn't really the case. Many different Linux *distributions* are available, each consisting of a Linux kernel along with a set of utilities and configuration files. The result is a complete OS, and two Linux distributions can differ from each other as much as either differs from macOS or even Windows. We therefore describe in more detail what a distribution is, what distributions are popular, and the ways in which distribution maintainers keep their offerings up-to-date.

Creating a Complete Linux-Based OS

Certification Objective

We've already described some of what makes up a Linux OS, but some details need reiteration or elaboration:

A Linux Kernel A Linux kernel is at the core of any Linux OS, of course. We've written this item as *a* Linux kernel because the Linux kernel is constantly evolving. Two distributions are likely to use slightly different kernels. Distribution maintainers also often *patch* kernels—that is, they make small changes to fix bugs or add features.

Core Unix Tools Tools such as the GNU tool set, the X Window System, and the utilities used to manage disks are critical to the normal functioning of a Linux system. Most Linux distributions include more or less the same set of such tools, but as with the kernel, they can vary in versions and patches.

Supplemental Software Additional software, such as major server programs, desktop environments, and productivity tools, ships with most Linux distributions. As with core Unix software, most Linux distributions provide similar options for such software. Distributions sometimes provide their own "branding," though, particularly in desktop environment graphics.

Startup Scripts Much of a Linux distribution's "personality" comes from the way it manages its startup process. Linux uses scripts and utilities to launch the dozens of programs that link the computer to a network, present a login prompt, and so on. These scripts and utilities vary between distributions, which means they have different features and may be configured in different ways.

An Installer Software must be installed to be used, and most Linux distributions provide unique installation software to help you manage this important task. Thus, two distributions may install in very different ways, giving you options for key features such as disk layouts and initial user account creation.

Typically, Linux distributions are available for download from their websites. You can usually download a CD-R or DVD image file that you can then burn to an optical disc. When you boot the resulting disc, the installer runs and you can install the OS. You can also use the same image to create a bootable USB flash drive if your computer lacks an optical drive. There are also cloud versions of many Linux distributions, which allow you to download a complete Linux system to run in either a private virtual machine or a commercial cloud such as Amazon Web Services (AWS) or Google Cloud Platform.

If you're curious about trying out Linux but don't have a dedicated workstation or laptop available, install a virtual machine software package such as VMWare or VirtualBox, which allows you to run a Linux OS inside an existing macOS or Windows workstation environment without having to change the system.

Some Linux installers come complete with all the software you're likely to install. Others come with only minimal software and expect you to have a working Internet connection so that the installer can download additional software. If your computer isn't connected to the Internet, be sure to get the right type of installer.

A Summary of Common Linux Distributions

Certification Objective

Depending on how you count, there are about a dozen major Linux distributions for desktop, laptop, and small server computers, and hundreds more that serve specialized purposes. Table 1.1 summarizes the features of the most important distributions.

TABLE 1.1 Features of major Linux distributions

Distribution	Availability	Package format	Release cycle	Administrator skill requirements
Arch	Free	pacman	Rolling	Expert
CentOS	Free	RPM	approximately 2-year	Intermediate
Debian	Free	Debian	2-year	Intermediate to expert
Fedora	Free	RPM	approximately 6-month	Intermediate
Gentoo	Free	ebuild	Rolling	Expert
Mint	Free	Debian	6-month	Novice to intermediate
openSUSE	Free	RPM	8-month	Intermediate
Red Hat Enterprise	Commercial	RPM	approximately 2-year	Intermediate
Slackware	Free	tarballs	Irregular	Expert
SUSE Enterprise	Commercial	RPM	2–3 years	Intermediate
Ubuntu	Free	Debian	6-month	Novice to intermediate

These features require explanation:

Availability Most Linux distributions are entirely open source or free software; however, some include proprietary components and are sold for money, typically with a support contract. Red Hat Enterprise Linux (RHEL) and SUSE Enterprise Linux are the two most prominent examples of this type of distribution. Both have completely free cousins. For RHEL, CentOS is a near-clone that omits the proprietary components, and Fedora is an open version that serves as a testbed for technologies that may eventually be included in RHEL. For SUSE Enterprise, openSUSE is a free alternative.

Package Format Most Linux distributions distribute software in *packages*, which are collections of many files in one. Package software maintains a local database of installed files, making upgrades and uninstallations easy. The RPM Package Manager (RPM) system is the most popular one in the Linux world, but Debian packages are very common, too. Other packaging systems work fine but are distribution-specific, such as the pacman package management system used in Arch Linux. Slackware is unusual in that it uses *tarballs* for its packages. These are package files created by the standard `tar` utility, which is used for backing up computers and for distributing source code, among other things. The tarballs that Slackware uses for its packages contain Slackware-specific information to help with package management. Gentoo is unusual because its package system is based on compiling most software from source code. This is time-consuming but enables experienced administrators to tweak compilation options to optimize the packages for their own hardware and software environments.

Tarballs are similar to the zip files common on Windows. Chapter 8, "Searching, Extracting, and Archiving Data," describes how to create and use tarballs.

Release Cycle We describe release cycles in more detail shortly, in "Understanding Release Cycles." As a general rule, distributions with short release cycles aim to provide the latest software possible, whereas those with longer release cycles strive to provide the most stable environments possible. Some try to have it both ways; for instance, Ubuntu releases long-term support (LTS) versions in April of even-numbered years. Its other releases aim to provide the latest software.

Administrator Skill Requirements The final column in Table 1.1 provides our personal estimation of the skill level required to administer a distribution. As you can see, we've described most Linux distributions as requiring "intermediate" skill to administer. Some, however, provide less in the way of user-friendly GUI administrative tools and so require more skill. Ubuntu aims to be particularly easy to use and administer.

Don't be scared off by the "intermediate" classification of most distributions. This book's purpose is to help you manage the essential features of such distributions.

Most Linux distributions are available for at least two platforms—that is, CPU types: x86 (also known as IA32, i386, and several variants) and x86-64 (also known as AMD64, EM64T, and x64). Until about 2007, x86 computers were the most common variety, but now x86-64 computers have become the standard. If you have an x86-64 computer, you can run either an x86 or an x86-64 distribution on it, although the latter provides a small speed improvement. More exotic platforms, such as ARM (for tablets), PowerPC, Alpha, and SPARC, are available. Such platforms are mostly restricted to servers and to specialized devices (described shortly).

Real World Scenario

Which Distribution to Use?

With a plethora of different Linux distributions available, one of the most often asked questions for novice Linux users is which one to try. Some Linux distributions, such as Red Hat and Oracle, focus on commercial Linux installations, providing paid customer support. These distributions can be expensive, and they are often hard for novice users to install and use.

A second type of Linux distributions are those geared toward advanced Linux users. The CentOS, Slackware, and Gentoo Linux distributions fall in this category. They expect you to know how to install and configure most of the software and hardware yourself. Advanced Linux users like to use these distributions because they can customize exactly what to install on the system.

The third type of Linux distributions are those geared toward novice Linux users. The popular Fedora, Ubuntu, and Mint Linux distributions fall in this category. The default installation takes care of most of the software and hardware configuration issues you need to worry about, and they all provide a wealth of user-friendly graphical tools for doing administrator functions, such as adding user accounts, exploring disk space, and working with network connections. These types of Linux distributions are the best way to go if you're a novice to the Linux world.

Understanding Release Cycles

Certification Objective

Table 1.1 summarized the release cycles employed by a number of common Linux distributions. The values cited in that table are the time between releases. For instance, new versions of Ubuntu come out every six months, like clockwork. Most other distributions' release schedules provide some "wiggle room"; if a release date slides a month, that may be acceptable.

After its release, a distribution is typically supported until sometime after the next version's release—typically a few months to a year or more. During this support period, the

distribution's maintainers provide software updates to fix bugs and security problems. After the support period has passed, you can continue to use a distribution, but you're on your own—if you need updated software, you'll have to compile it from source code yourself or hope that you can find a compatible binary package from some other source. As a practical matter, therefore, it's generally a good idea to upgrade to the latest version before the support period ends. This fact makes distributions with longer release cycles appealing to businesses, since a longer time between installations minimizes disruptions and costs associated with upgrades.

Two of the distributions in Table 1.1 (Arch and Gentoo) have rolling release cycles. Such distributions have no version numbers in the usual sense; instead, upgrades occur in an ongoing manner. Using such a distribution makes it unnecessary to ever do a full upgrade, with all the hassles that creates; however, you'll occasionally have to do a disruptive upgrade of one particular subsystem, such as a major upgrade in your desktop environment.

Certification Objective

Before the release of a new version, most distributions make pre-release versions available. *Alpha software* is extremely new and very likely to contain serious bugs, whereas *beta software* is more stable but nonetheless more likely to contain bugs than is the final release software. As a general rule, you should avoid using such software unless you want to contribute to the development effort by reporting bugs or unless you're desperate to have a new feature.

Embedded Linux Systems

In addition to the mainstream PC distributions, several other Linux distributions are available that serve more specialized purposes. The term *embedded systems* describes running a small stripped-down Linux system on a small microcomputer, such as a phone or monitoring device. Common uses of embedded Linux systems are:

Certification Objective

Android Many cell phones today use a Linux-based OS known as *Android*. Its user interface is similar to that of other smartphones, but underneath lies a Linux kernel and a significant amount of the same Linux infrastructure you'll find on a PC. Such phones don't use X or typical desktop applications, though; instead, they run specialized applications for cell phones.

Android is best known as a cell phone OS, but it can be used on other devices. Some tablets and e-book readers, for instance, run Android.

Network Appliances Many broadband routers, print servers, and other devices you plug into a local network to perform specialized tasks run Linux. You can sometimes replace

the standard OS with a customized one if you want to add features to the device. Tomato (`www.polarcloud.com/tomato`) and OpenWrt (`openwrt.org`) are two examples of such customized Linux distributions. Don't install such software on a whim, though; if done improperly, or on the wrong device, they can render the device useless!

IoT Devices In recent years the term *Internet of Things (IoT)* has exploded both in the news and in classrooms. IoT relates to creating a network of small devices that can sense physical conditions and control systems. Small microprocessors running a specialized OS monitor data from sensors, such as the temperature, humidity, light, or motion, and use that data to control motors, locks, and switches. Embedded controllers such as the Arduino and Beagle Bones devices, as well as more powerful larger controllers such as the Raspberry Pi, are becoming all the craze in schools and manufacturing environments. The larger controllers, such as the Raspberry Pi, often run a stripped-down version of Linux to provide more versatility in the applications, such as sending out email alerts for specific sensor conditions.

TiVo This popular digital video recorder (DVR) uses a Linux kernel and a significant number of standard support programs, along with proprietary drivers and DVR software. Although many people who use them don't realize it, they are Linux-based computers under the surface.

Most embedded Linux systems typically require little or no administrative work from users, at least not in the way such tasks are described in this book. Instead, these devices have fixed basic configurations and guided setup tools to help inexperienced users set critical basic options, such as network settings and your time zone.

Linux in the Cloud

Cloud technology has greatly changed the landscape of the computer world. Moving computer resources and applications into a shared network environment changes how many companies do business and provide services to customers. Linux plays an important role in the cloud world, so it's a good idea to define just what a cloud is and what type of resources it provides. This section covers the basics of cloud computing.

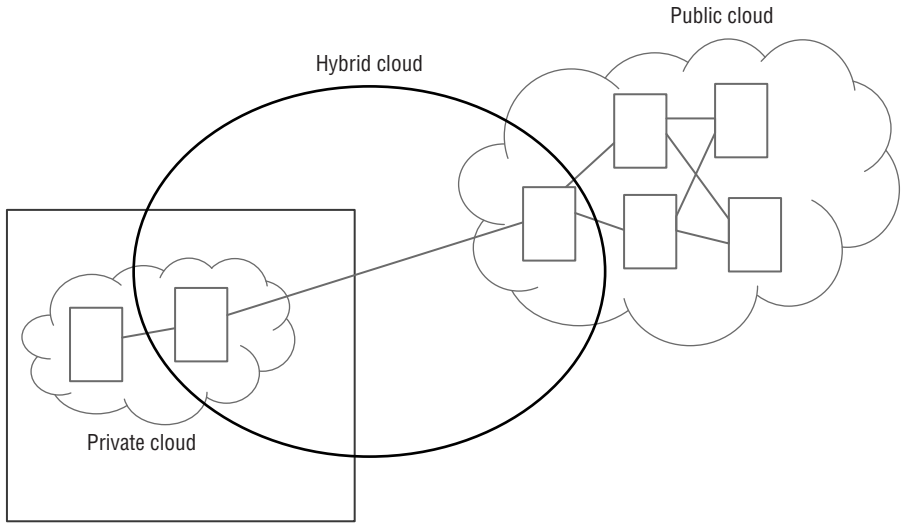
What Is Cloud Computing?

The first mention of the term *cloud* came in documentation for the original ARPANET network environment in 1977, the precursor to the modern-day Internet. In that documentation, the cloud symbol was commonly used to represent the large network of interconnected servers geographically dispersed. However, in this environment each server was self-contained and self-sufficient—there was no distributed computing.

The term *cloud computing* is related to distributed computing. In distributed computing, resources are shared among two or more servers to accomplish a single task, such as run an application. This environment became the precursor to what we know today as cloud computing, popularized by companies such as Amazon Web Services (AWS), Google Cloud Platform, and Microsoft Azure.

Cloud computing provides the ability to deliver computing resources across the Internet. Now customers can purchase both hardware and software resources as needed from cloud computing vendors. This includes servers, storage space, databases, networks, operating systems, and even individual applications. Figure 1.4 demonstrates the three methods for providing cloud computing services.

FIGURE 1.4 Cloud computing methods



As you can see in Figure 1.4, there are three primary methods for providing cloud computing environments:

- **Public**: In the public cloud computing environments, a third party provides all of the computing resources outside of the organization. These resources are usually shared between multiple organizations.
- **Private**: In the private cloud computing environments, each individual organization builds its own cloud computing resources to provide resources internally.
- **Hybrid**: In hybrid cloud computing environments, computing resources are provided internally within the organization but also connected to an external public cloud to help supplement resources when needed.

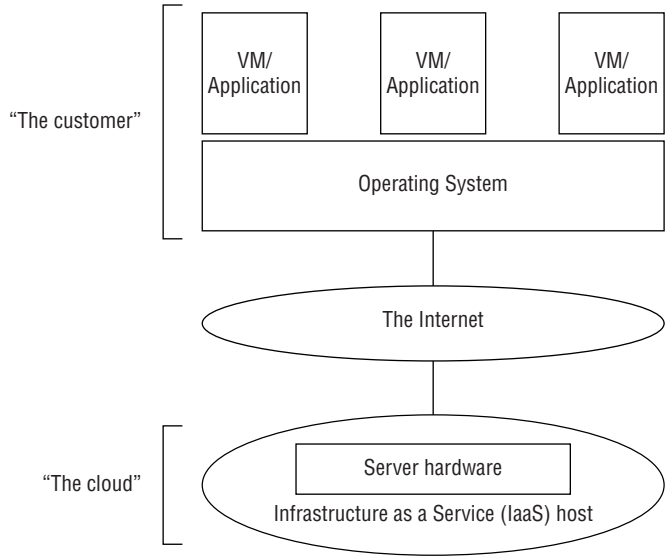
What Are the Cloud Services?

Cloud computing environments can customize the level of resources provided to customers, depending on each customer's needs. This section describes the three most popular models for providing resource levels that you'll find from cloud computing vendors.

Infrastructure as a Service (IaaS)

In the infrastructure as a service (IaaS) model, the cloud computing vendor provides low-level server resources to host applications for organizations. These low-level resources include all of the physical components you'd need for a physical server, including CPU time, memory space, storage space, and network resources, as shown in Figure 1.5.

FIGURE 1.5 The IaaS cloud model



The server resources provided may be on a single server, or they may be distributed among several servers. In a distributed environment. the servers may be co-located in a single facility or they may be separated into multiple facilities located in separate cities. This helps provide for increased availability.

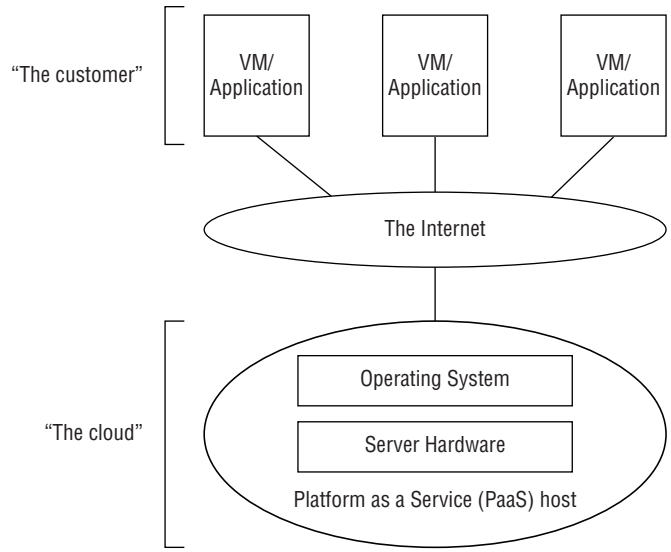
As shown in Figure 1.5, in an IaaS model the customer supplies the operating system and any applications that it needs to run. Most IaaS environments support a multitude of different operating systems, including Linux and Windows servers. The customer is responsible

for any system administration work required for the operating system, as well as any application administration. The cloud computing vendor maintains the physical infrastructure environment.

Platform as a Service (PaaS)

In the platform as a service (PaaS) model, the cloud computing vendor provides both the physical server environment as well as the operating system environment to the customer, as shown in Figure 1.6.

FIGURE 1.6 The PaaS cloud model

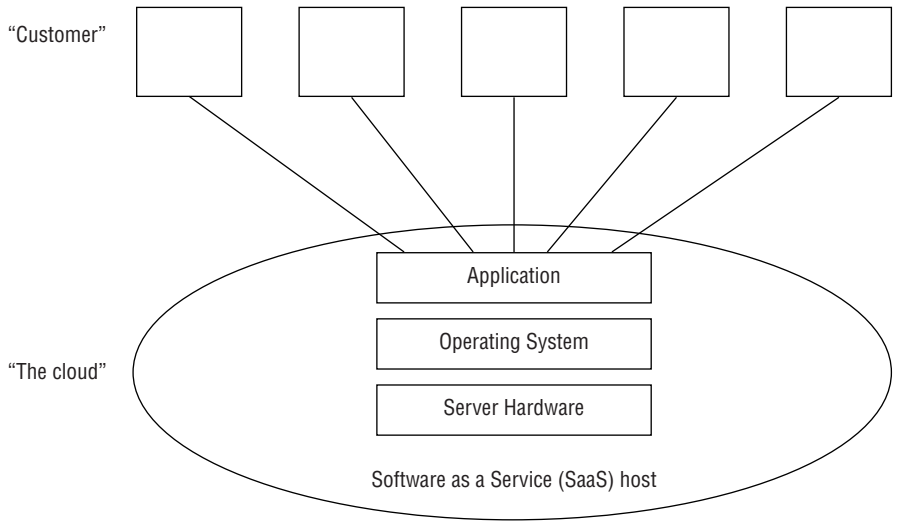


With the PaaS model. the cloud computing vendor takes responsibility for both the physical components as well as the operating system administration. It provides system administration support to ensure the operating system is properly patched and updated to keep up with current releases and security features. This allows the customer to focus mainly on developing the applications running within the PaaS environment.

Software as a Service (SaaS)

In the software as a service (SaaS) model, the cloud computing vendor provides a complete application environment, such as a mail server, database server, or web server. The vendor provides the physical server environment, the operating system, and the application software necessary to perform the function. This is shown in Figure 1.7.

FIGURE 1.7 The SaaS cloud model



Summary

Linux is a powerful OS that you can use on everything from a cell phone to a supercomputer. At Linux's core is its kernel, which manages the computer's hardware. Built atop that are various utilities (many from the GNU project) and user applications. Linux is a clone of the Unix OS, with which it shares many programs. Apple macOS is another Unix OS, although one with a unique user interface. Although Windows shares many features with Unix, it's an entirely different OS, so software compatibility between Linux and Windows is limited. Linux comes in many varieties, known as distributions, each of which has its own unique "flavor." Because of this variety, you can pick a Linux version that best suits your needs, based on its ease of use, release cycle, and other unique features. Linux is a popular OS used for embedded systems, due to its customizable features and small size. The same customizable features also makes it ideal for large cloud computing environments that support applications distributed around the world.

Exam Essentials

Describe what makes a Linux distribution and why there are so many. A Linux distribution is a bundle of components required to run a Linux system. This includes a Linux kernel and usually utilities for managing the system, application software, and a package

management system for installing and removing software. Different Linux users have different needs, such as desktop office automation, multimedia production, mathematical simulations, or server features, such as a web or database server. The different Linux distributions are each customized to support specific features and functions so that you don't have to do that yourself.

Explain how Linux is used in embedded systems. Embedded systems often use a stripped-down Linux system that specializes in controlling specific hardware. Devices such as Android phones, IoT monitors, and TiVo recorders each use a customized Linux system that performs only the functions required for those devices.

Explain how Linux is used in cloud environments. The cloud environment requires a distributed computing environment that can be expanded as needed. Linux servers provide an inexpensive platform that can be easily modified.

Describe the basic differences between Linux and the more popular macOS and Windows environments. The basic difference between Linux and the macOS and Windows environments is choice. For just about every feature of the OS, Linux doesn't lock you in to a specific environment, but instead provides multiple options for you to choose from. This includes desktop features, application software, and even what you pay for software support.

Explain Linux distribution life-cycle management. Each OS requires updating from time to time to keep up with technology, improve features, fix software bugs, and guard against security vulnerabilities. Life-cycle management relates to how often an OS is updated. Some OSs are updated on a regular basis, whereas others are updated only as needed. Different Linux distributions support different life cycles, depending on their user base. Some Linux distributions produce long-term support (LTS) versions that are maintained for three to five years, ideal for corporate environments that don't want to change OS versions for hundreds of thousands of employees.

Review Questions

You can find the answers in the Appendix A.

1. Which of the following is a function of the Linux kernel? (Choose all that apply.)
   - **A.** Allocating memory for use by programs
   - **B.** Allocating CPU time for use by programs
   - **C.** Creating menus in GUI programs
   - **D.** Controlling access to hard disks
   - **E.** Enabling programs to use the network
2. Which of the following is an example of an embedded OS?
   - **A.** Android
   - **B.** CentOS
   - **C.** Fedora
   - **D.** Mint
   - **E.** Red Hat
3. Which of the following is a notable difference between Linux and macOS?
   - **A.** Linux can run common GNU programs, whereas macOS cannot.
   - **B.** Linux's GUI is based on the X Window System, whereas macOS is not.
   - **C.** Linux cannot run on Apple Macintosh hardware, whereas macOS can run only on Apple hardware.
   - **D.** Linux relies heavily on BSD software, whereas macOS uses no BSD software.
   - **E.** Linux supports text-mode commands, whereas macOS is a GUI-only OS.
4. Where did the Linux kernel come from?
   - **A.** It was derived from Microsoft Windows.
   - **B.** It was derived from Apple macOS.
   - **C.** It was derived from AT&T Unix.
   - **D.** It was derived from BSD Unix.
   - **E.** It was created by Linus Torvalds.
5. True or false: If you log into a Linux system in graphical mode, you cannot use text-mode commands in that session.
6. True or false: CentOS is a Linux distribution with a long release cycle.

7. A Linux text-mode login prompt reads _____.
   - **A.** login:
   - **B.** welcome:
   - **C.** Enter:
   - **D.** userid:
   - **E.** Enter your userid:

8. A common security problem with Windows that's essentially nonexistent on Linux is _____.
   - **A.** Commercial software
   - **B.** Network firewalls
   - **C.** Network routers
   - **D.** Viruses
   - **E.** Software management packages

9. Pre-release software that's likely to contain bugs is known as _____ and _____.
   - **A.** First and second
   - **B.** Primary and secondary
   - **C.** Alpha and beta
   - **D.** Development and production
   - **E.** Development and test

10. Linux distributions that have no version number but instead release upgrades in an ongoing manner are said to have a(n) _____ release.
    - **A.** rolling
    - **B.** staggered
    - **C.** informal
    - **D.** systematic
    - **E.** hap-hazard